AN OBJECT-ORIENTED INTERFACE TO THE CCSDS GROUND TELECOMMAND SERVICES

Tim Ray, NASA
Jeff Condron, Raytheon
Goddard Space Flight Center, NASA
Greenbelt, MD, USA 20771
Timothy.J.Ray.1@gsfc.nasa.gov
Jeffrey.Condron@gsfc.nasa.gov

Abstract

The Telecommand Data Routing and Channel Services defined by the Consultative Committee for Space Data Systems (CCSDS) are flexible enough to support a myriad of commanding models. Because the standard is so broad, the traditional ground system approach has been to implement only the portion of the standard needed by the particular spacecraft being tested/operated.

Tasked with providing ground Telecommand Services for an entire class of spacecraft, where each spacecraft may choose any valid CCSDS commanding model, we designed an interface capable of supporting the full CCSDS protocol.

Significant cost savings are achieved by using the same interface and implementation for multiple spacecraft.

This paper describes this interface and how it leads to a straightforward implementation. The concepts used are general enough to be applied to other problems.

Keywords: CCSDS Telecommand Services, Ground Systems, SFDU.

CCSDS Telecommand Services

The purpose of the ground Telecommand Services is to reliably deliver your commands to the spacecraft. This is accomplished via a series of layers which add headers and trailers to your command Packet and pass the resulting data to the spacecraft.

The flight Telecommand Service layers then remove the headers and trailers to extract the original command Packet.

Each ground layer accepts a specific form of input data and generates a specific form of output data, as follows:

Table 1: Layers of the Telecommand Services

Layer	Input data	Output data
Segmentation	Packet	Segment
Transfer	Segment	Transfer Frame
Coding	Transfer Frame	Command Link Transmission Unit (CLTU)
Physical	CLTU	command data for the spacecraft

The Telecommand Services are quite flexible. They allow:

1. *data entry at user-specified layers:*

   The services allow you to insert data at either the Segmentation Layer or Transfer Layer, while the testing of flight command hardware/software requires data entry at the Coding and Physical Layers. Therefore, a generic implementation of the standard should allow data entry at any layer of the services.

2. *commands to be sent to the spacecraft individually or in groups:*

   The services allow you to group commands by putting multiple Packets in one Segment and/or multiple Transfer Frames in one CLTU.

3. *configuration of each layer:*

   Each layer of the services provides a number of user-definable parameters, which must be set to match the flight layers used on your particular spacecraft.

Interface Requirements

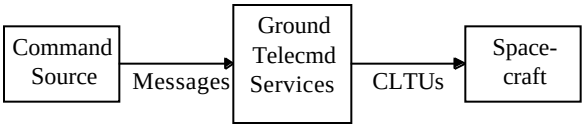


Figure 1: Command Data Flow

As described above, to send commands to the spacecraft, you send messages to the Telecommand Services.

The Telecommand Services convert the given data to a form the spacecraft can understand, and the resulting data is transmitted to the spacecraft.

The format of these messages must support communication of the following information:

- the command data you want sent and what form it is in (e.g. packet, segment)
- how you want the data grouped (e.g. multiple packets in one segment)
- how each layer of the Telecommand Services should be configured to speak to your spacecraft.

Interface Goals

An important goal is to provide a simple interface to the Telecommand Services. This is challenging because there are many forms of data to communicate.

The message format is your interface to the Telecommand Services. Thus, it should meet some of the same goals any good user interface would. Namely:

- messages should be easy to generate and read;
- all messages should have a consistent format, and be self-defining;
- messages should be available for all functions that you may need to perform;
- messages should indicate the action to be taken and contain the data to take it upon;
- the system should be expandable, meaning additional message types can be added without modifying existing functions and formats;
- feedback should be given to each message indicating success or failure.

The solution–concepts

Interface concepts (objects)

We chose to base our message format on another CCSDS standard, the Standard Formatted Data Unit (SFDU). Some characteristics of SFDUs:

1. There are several classes of SFDUs, including I-class (we'll call these envelopes) and Z-class (we'll call these mailbags).
2. Envelopes always contain data.
3. Mailbags may contain envelope(s) and/or other mailbag(s).
4. Each mailbag/envelope is labeled with a four character string describing it's contents.

Our messages each contain one or more objects; each object is an SFDU. Some characteristics of these objects are:

1. There are 2 types of objects: envelopes and mailbags.
2. An envelope contains one piece of data to be acted upon.
3. A mailbag contains a group of envelopes and/or mailbags, (i.e. mailbags are used to combine multiple input data items into a single output data item)
4. The label describes the contents of an object (e.g. whether it is a packet, segment, or whatever), and specifies the actions to be taken with the data.

Implementation concepts (message conversion)

When you send a message to the Telecommand Services, it is passed through each layer. When a layer receives a message, it applies the following rules to each mailbag and envelope within the message and then returns the resulting message.

1. If the label on an envelope indicates input data for this layer, then the envelope is replaced with an envelope containing output data from this layer (built from the contents of the input envelope). For example, the Segmentation Layer replaces each Command Packet envelope with a Command Segment envelope.

2. If the label on a mailbag indicates grouping to be performed by this layer, then the mailbag is replaced with an envelope containing output data from this layer (built from the contents of the input mailbag). For example, the Segmentation Layer replaces a Segment mailbag with a Segment envelope (all the packets within the mailbag are combined into one Segment).

3. If the label on an envelope indicates a Configuration Directive for this layer, then the envelope is removed and the directive is executed.

4. Any mailbag/envelope whose label is unrecognized is left in the message. This is a key characteristic of the implementation because it ensures that objects (i.e. mailbags and envelopes) pass through the services until they reach the applicable layer. It also ensures that the existing implementation is not affected when new objects are defined. Note: All envelopes are checked, including those within unrecognized mailbags.

After the message has passed through each layer, the data inside will have been transformed appropriately to a form that your spacecraft can understand. The resulting data is then transmitted to your spacecraft.

The solution–specifics

List of objects

This section lists the objects that a Command Source may include in messages sent to our Telecommand Services. The following are defined:

Table 2: Envelope Objects

Label	Meaning
DSEG	configuration directive for the Segmentation Layer.
DXFR	configuration directive for the Transfer Layer.
DCOD	configuration directive for the Coding Layer.
DPHY	configuration directive for the Physical Layer.
CPKT	Command Packet data
CSEG	Command Segment data
CFRM	Command Transfer Frame data
CLTU	Command Link Transmission Unit data
CSPA	Command data for the spacecraft

Table 3: Mailbag Objects

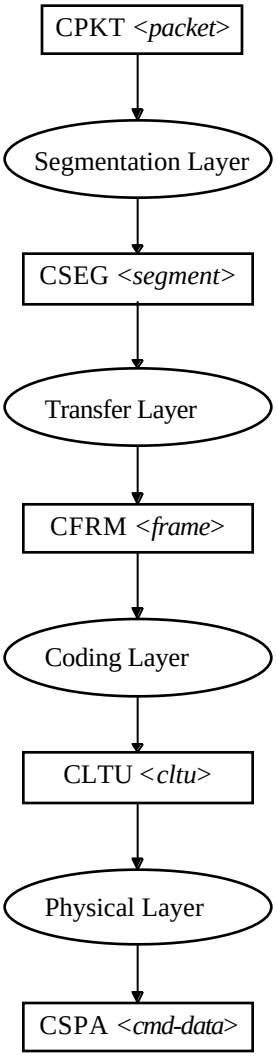
Label	Meaning
CSEG	build one Command Segment from all the data in this mailbag.
CLTU	build one CLTU from all the data in this mailbag.

Commanding examples:

A) *Data entry at each layer of the services:*

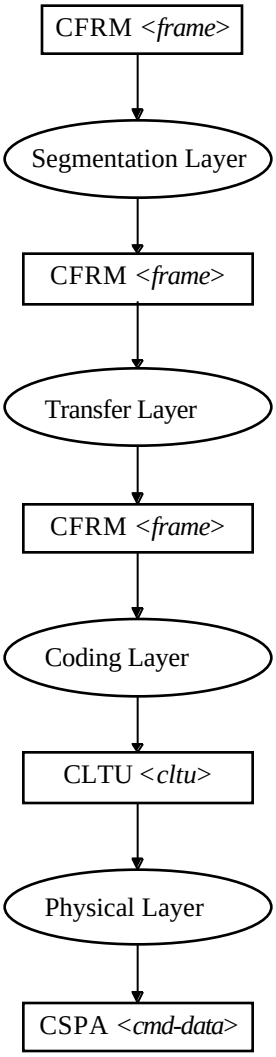
Envelopes containing command data are passed through the Telecommand Services until they reach the applicable layer; from then on they get converted.

Example 1 - Sending a Packet to the spacecraft:



The Packet envelope is recognized at the Segmentation Layer, so headers/trailers are added by every layer.

Example 2 - Sending a Frame to the spacecraft:

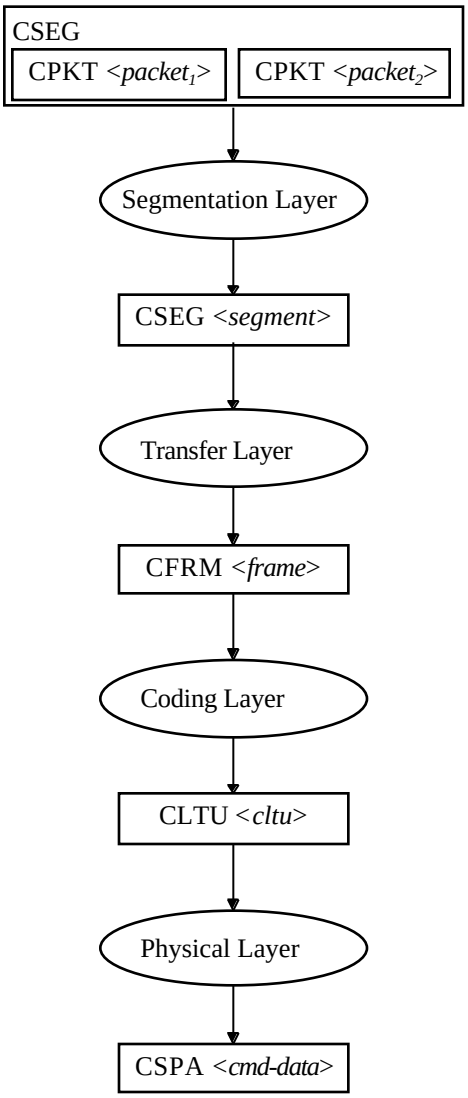


The Frame envelope is not recognized until it reaches the Coding Layer, so headers/trailers are added beginning with the Coding Layer. Note that the Segmentation and Transfer Layers do not modify the message at all.

B) Data aggregation:

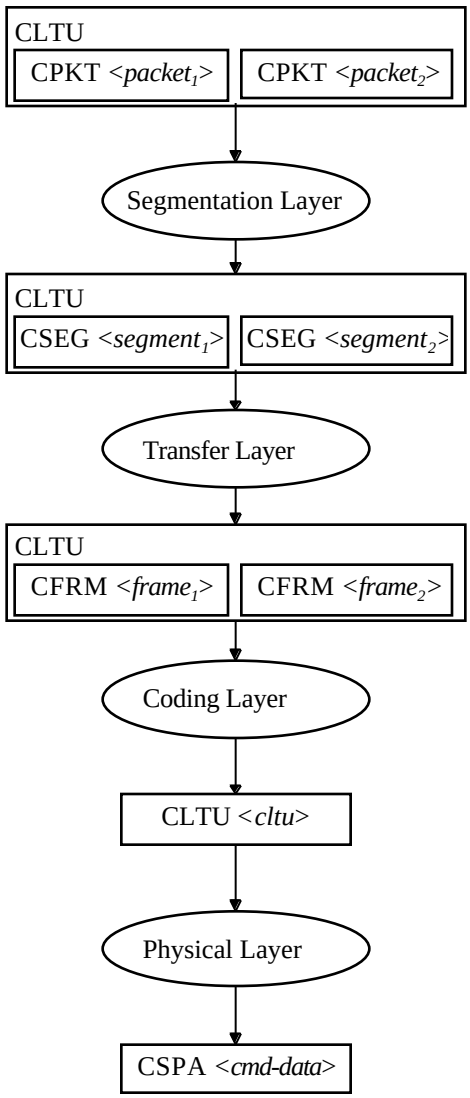
Data aggregation is requested by putting envelopes in a mailbag. The mailbag remains in the message until it reaches the applicable layer; envelopes within mailbags are converted in the normal way by each layer.

Example 3– 2 Packets into 1 Segment:


CSEG
CPKT <packet1>
CPKT <packet2>
Segmentation Layer
CSEG
Transfer Layer
CFRM <frame>
Coding Layer
CLTU <cltu>
Physical Layer
CSPA <cmd-data>

The Segmentation Layer recognizes the Segment mailbag, so it replaces the entire mailbag with a Segment envelope containing the data from both command packets (i.e. both packets are combined into 1 Segment). Headers/trailers are added in the usual way by the remaining layers.

Example 4– 2 Transfer Frames into 1 CLTU:



The Segmentation Layer does not recognize the CLTU mailbag, but it does recognize the Packet envelopes inside, so it converts each Packet envelope to a Segment envelope.

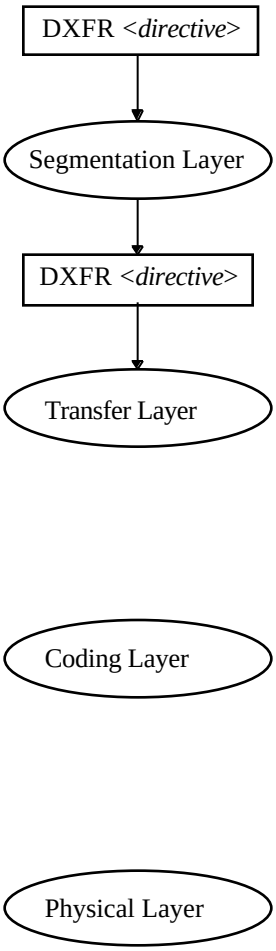
The Transfer Layer doesn't recognize the CLTU mailbag either, but it does recognize the Segment envelopes inside, so it converts each Segment into a Frame.

The Coding Layer recognizes the CLTU mailbag, so it replaces the entire mailbag with a single CLTU envelope containing the data from both Frames (i.e. 2 Frames are combined into 1 CLTU). Headers/trailers are added in the usual way by the remaining layer.

C) Configuration of each layer:

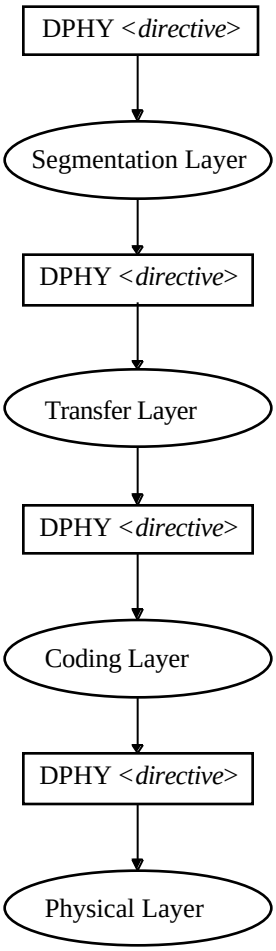
Envelopes containing configuration directives are passed through the Telecommand Services until they reach the applicable layer; at that point they get executed and removed.

Example 5 - A configuration directive for the Transfer Layer



The Transfer-layer directive envelope is not recognized until it reaches the Transfer Layer, where the directive is executed and the envelope is removed.

Example 6 - A configuration directive for the Physical Layer



The Physical-layer directive envelope is not recognized until it reaches the Physical Layer, where the directive is executed and the envelope is removed.

Responses to the Command Source:

When you send messages to the services, you receive responses indicating:

1. if your message was accepted or rejected, and
2. whether or not your command(s) were successfully delivered to the spacecraft.

In writing this paper, we felt the concepts were more clearly illustrated if the response mechanism is not included.

The solution–characteristics

Characteristics of the interface

Flexible: A single interface can handle any valid CCSDS commanding model.

Object-oriented: Each message from the command source specifies both the input data (e.g. Command Packets) and the actions to be performed with the data (e.g. to combine multiple packets into one segment).

Expandable: New capabilities are added by defining new objects (SFDUs). Adding new objects does not affect the existing implementation, because objects pass through the services until they reach the applicable layer. This is important, since additional layers (outside the Telecommand Services) are needed to communicate with different Ground Stations.

Characteristics of the implementation

Modular: The logic for each service layer is contained in a separate Application Program Interface (API). The objects used for the Command Source interface are also used for communication between layers.

Distributable: The design can be split along any layer boundary for distribution across multiple platforms. Because the objects represent data in ASCII format, there is data consistency across platforms.

Conclusion

By choosing an elegant interface, this complex problem was solved with a straightforward implementation. The concepts we used in this interface are general enough to be used in solving other problems.

The resulting implementation is currently being used by NASA (National Aeronautics and Space Administration) for integration & test of the Microwave Anisotropy Probe (MAP) and Earth Observer-1 (EO-1) spacecraft. It will be used for post-launch operations of these spacecraft as well as the Imager for Magnetopause to Aurora Global Exploration (IMAGE) spacecraft. Use of the same implementation and interface across multiple spacecraft has yielded significant cost savings.